

Randomized Linear Programming Solves the Discounted Markov Decision Problem In Nearly-Linear Running Time

Mengdi Wang

Department of Operations Research and Financial Engineering, Princeton University, Princeton, NJ
email: mengdiw@princeton.edu

April 22, 2017

Abstract

We propose a randomized linear programming algorithm for approximating the optimal policy of the discounted Markov decision problem. By leveraging the value-policy duality, the algorithm adaptively samples state transitions and makes exponentiated primal-dual updates. We show that it finds an ϵ -optimal policy using nearly-linear running time in the worst case. For Markov decision processes that are ergodic under every stationary policy, we show that the algorithm finds an ϵ -optimal policy using running time linear in the total number of state-action pairs, which is sublinear in the input size. These results provide new complexity benchmarks for solving stochastic dynamic programs.

Keywords: Markov decision process, randomized algorithm, linear programming, duality, primal-dual method, running-time complexity, stochastic approximation

1 Introduction

Markov decision process (MDP) is a fundamental model for sequential decision-making problems in dynamic and random environments. It models a stochastic control process in which a planner aims to make a sequence of decisions as the state of the process evolves. MDP serves as the basic mathematical framework for dynamic programming, stochastic control and reinforcement learning. It is widely applied in engineering systems, artificial intelligence, e-commerce and finance.

We focus on the Discounted Markov Decision Problem (DMDP) in which one aims to make an infinite sequence of decisions and optimize some cumulative sum of discounted rewards. An instance of the DMDP can be described by a tuple $\mathcal{M} = (\mathcal{S}, \mathcal{A}, \mathcal{P}, \mathbf{r}, \gamma)$, where \mathcal{S} is a finite state space of size $|\mathcal{S}|$, \mathcal{A} is a finite action space of size $|\mathcal{A}|$, $\gamma \in (0, 1)$ is a discount factor, \mathcal{P} is the collection of state-to-state transition probabilities $\mathcal{P} = \{p_{ij}(a) \mid i, j \in \mathcal{S}, a \in \mathcal{A}\}$, \mathbf{r} is the collection of state transitional rewards $\mathbf{r} = \{r_{ij}(a) \mid i, j \in \mathcal{S}, a \in \mathcal{A}\}$ where $r_{ij}(a) \in [\frac{1}{2}, 1]$. We also denote by \mathbf{r}_a the vector of expected state-transition reward under action a , where $\mathbf{r}_{a,i} = \sum_{j \in \mathcal{S}} p_{ij}(a) r_{ij}(a)$. Suppose that the decision process is in state i , if action a is selected, the process moves to a next state j with probability $p_{ij}(a)$ and generates a reward $r_{ij}(a)$. The input dimension of the DMDP tuple \mathcal{M} is $\mathcal{O}(|\mathcal{S}|^2 |\mathcal{A}|)$.

Our goal is to find the best sequence of actions to choose at all possible states in order to maximize the expectation of the cumulative reward. More precisely, we want to find a (stationary) policy that specifies which action to choose at each state. A stationary and randomized policy can be represented by a collection of probability distributions $\pi = \{\pi_i\}_{i \in \mathcal{S}}$, where $\pi_i : \mathcal{A} \mapsto [0, 1]$ is a

vector of probability distribution over actions. We denote by P^π the transition probability matrix of the DMDP under a fixed policy π , where $P_{ij}^\pi = \sum_{a \in \mathcal{A}} \pi_i(a) p_{ij}(a)$ for all $i, j \in \mathcal{S}$. The objective of the DMDP is to find an optimal policy π^* such that the infinite-horizon sum of discounted rewards is maximized:

$$\max_{\pi} \mathbf{E}^\pi \left[\sum_{t=1}^{\infty} \gamma^t r_{i_t i_{t+1}}(a_t) \right],$$

where $\{i_0, a_0, i_1, a_1, \dots, i_t, a_t, \dots\}$ are state-action transitions generated by the Markov decision process under the fixed policy π , and the expectation $\mathbf{E}^\pi[\cdot]$ is taken over the entire trajectory. In total there are $|\mathcal{A}|^{|\mathcal{S}|}$ distinct deterministic policies.

Despite its strong power of modeling, MDP is generally considered as a difficult problem due to the *curse of dimensionality*. There have been tremendous efforts in analyzing the complexity of MDP and its solution methods. Most existing studies focus on deterministic methods that find the exact optimal policy. Due to the curse of dimensionality, finding the exact optimal policy is often prohibitively difficult, especially in large-scale applications such as computer games and robotics. We ask the following question: Is there a way to trade the precision of the exact optimal policy for a better time complexity? Motivated by this question, we are interested in developing randomized algorithms that can approximate the optimal policy efficiently. In particular, we are interested in reducing the complexity's dependence on sizes of the state and action spaces.

In this paper, we develop a novel randomized linear programming method for solving the DMDP. The method is motivated from the linear programming formulation of the nonlinear Bellman equation and the linear duality between the value and policy functions. It has two main components. One component of the method is adaptive simulation of state-to-state transitions of the Markov decision process. The other component is a stochastic primal-dual iteration that applies to a specially constructed saddle point problem. The dual update step uses projection onto an information set with respect to a special Bregman divergence function. The information set and the Bregman divergence function are carefully crafted to fully utilize properties of the DMDP.

We analyze the algorithm's complexity for obtaining a (relatively) ϵ -optimal policy. We focus on its running-time complexity in terms of the total number of arithmetic operations, which include addition, subtraction, multiplication, division and exponentiation. We use $\mathcal{O}(1)$ to denote some absolute constant number, and we use $\tilde{\mathcal{O}}(1)$ to hide polylog $\left(|\mathcal{S}|, |\mathcal{A}|, \frac{1}{\epsilon}, \frac{1}{1-\gamma}\right)$ factors. Our main results are summarized as follows:

1. We show that there exists an algorithm that finds an ϵ -optimal policy $\hat{\pi}$ with probability at least $1 - \delta$ in running time

$$\tilde{\mathcal{O}} \left(\frac{|\mathcal{S}|^3 |\mathcal{A}|}{(1-\gamma)^4 \epsilon^2} \log \left(\frac{1}{\delta} \right) \right).$$

We recall that the input dimension of the DMDP is $\mathcal{O}(|\mathcal{S}|^2 |\mathcal{A}|)$. Therefore, the algorithm achieves a *linearly-linear running-time* complexity.

2. In the case where the Markov decision process is known to be ergodic under any stationary policy, we show that the algorithm finds an ϵ -optimal policy with probability at least $1 - \delta$ in running time

$$\tilde{\mathcal{O}} \left(|\mathcal{S}|^2 |\mathcal{A}| + \frac{|\mathcal{S}| |\mathcal{A}|}{(1-\gamma)^2 \epsilon^2} \log \left(\frac{1}{\delta} \right) \right).$$

The first term $\tilde{\mathcal{O}}(|\mathcal{S}|^2 |\mathcal{A}|)$ is due to an initialization step that preprocesses the input data (consisting mainly of transition probabilities) into a sampler. The preprocessing step can be omitted or expedited if the input data are given in a suitable data structure that can be

directly used as a sampler (e.g., sorted arrays or binary trees). When the state space \mathcal{S} is very large and ϵ is moderate, we have $\frac{|\mathcal{S}||\mathcal{A}|}{(1-\gamma)^2\epsilon^2} \ll |\mathcal{S}|^2|\mathcal{A}|$. In this case, we obtain a surprising result of *sublinear running-time complexity*. In other words, it is possible to obtain a near-optimal policy without even reading all entries of input data.

The preceding complexity results draw a sharp contrast with the best-known complexity results for policy iteration, value iteration, and interior point methods for solving the DMDP (see Table 1 for a detailed comparison). We establish the first nearly-linear running-time complexity for solving arbitrary DMDP and the first sublinear complexity for solving ergodic DMDP. The latter complexity result is linear in the total number of state-action pairs, which we conjecture to be non-improvable in its dependence on $|\mathcal{S}|$ and $|\mathcal{A}|$. The new method uses simulated state transitions, which makes it potentially amenable to online learning settings.

Notations All vectors are considered as column vectors. For a vector $\mathbf{x} \in \mathbb{R}^n$, we denote by x_i or $x(i)$ its i -th component, denote by \mathbf{x}^\top its transpose, and denote by $\|\mathbf{x}\| = \sqrt{\mathbf{x}^\top \mathbf{x}}$ its Euclidean norm. We denote by $\mathbf{e} = (1, \dots, 1)^\top$ the vector with all entries equaling 1, and we denote by \mathbf{e}_i the vector with its i -th entry equaling 1 and other entries equaling 0. For a positive number x , we denote by $\log x$ the natural logarithm of x . For two probability distributions p, q over a finite set X , we denote by $D_{KL}(p||q)$ their Kullback-Leibler divergence, i.e., $D_{KL}(p||q) = \sum_{x \in X} p(x) \log \frac{p(x)}{q(x)}$.

2 Related Literatures

There are three major approaches for solving MDPs: the value iteration method, the policy iteration method, and linear programming methods. For more detailed surveys on MDP and its solution methods, we refer the readers to the textbooks [3, 5, 24, 4] and references therein.

The value iteration is developed by Bellman [1] as an successive approximation method to solve nonlinear fixed-point equations. There have been many efforts in analyzing the convergence and complexity of value iteration; see for examples [26, 20]. The best known complexity result for the value iteration is $|\mathcal{S}|^2|\mathcal{A}|L \frac{\log(1/(1-\gamma))}{1-\gamma}$, where L is the number of bits to represent the input ($L \geq |\mathcal{S}|^2|\mathcal{A}|$). Later it was showed by [16] that the value iteration method is not strongly polynomial for DMDP. The first development of policy iteration is by Howard [17]. The complexity of policy iteration has been analyzed extensively; see for examples [21, 30, 25] and references therein.

Not long after the development of value and policy iterations, [15] and [14] discovered that the Bellman equation can be formulated into an equivalent linear program. It followed that one can apply linear programming method such as the simplex method by Dantzig [12] to solve MDP. Later [29] showed that a combinatorial interior-point algorithm (CIPA) solves the DMDP in strongly polynomial time. The seminal work by Ye [30] showed that the policy iteration of discounted MDP is a form of dual simplex method and terminates in $\mathcal{O}(\frac{S^2 A}{1-\gamma} \log(\frac{|\mathcal{S}|}{1-\gamma}))$ number of iterations. More importantly, it shows that the policy iteration and the simplex method are strongly polynomial for the DMDP, which means that the time complexity is determined by only the input dimension rather than size of the input values. Later [25] improved the result by removing the $\log |\mathcal{S}|$ factor. Recent development [19] showed that linear programs can be solved in $\tilde{\mathcal{O}}(\sqrt{\text{rank}(A)})$ number of linear system solves, which leads to running time $\tilde{\mathcal{O}}((|\mathcal{S}||\mathcal{A}| + |\mathcal{S}|^{2.3729})\sqrt{|\mathcal{S}|})$ for solving the DMDP. We note that all the aforementioned methods, i.e., value iteration, policy (and simplex) iteration, linear programming methods, require solving multiple linear systems of equations. Each equation solve requires up to $\mathcal{O}(|\mathcal{S}|^2|\mathcal{A}|)$ arithmetic operations.

Let us summarize the worst-case running-time complexity of solution methods for DMDP in Table 1. The running-time complexity is in terms of the total number of arithmetic operations. According to Table 1, the worst-case complexity of policy iteration is quadratic with respect to the input size $|\mathcal{S}|^2|\mathcal{A}|$. In comparison, our proposed randomized linear programming method achieves a running-time complexity that is linearly linear (sometimes even sublinear) with respect to the input dimension. The complexity of the our new method has the sharpest dependence on the input dimensions $|\mathcal{S}|$ and $|\mathcal{A}|$.

Value Iteration	$ \mathcal{S} ^2 \mathcal{A} L^{\frac{\log(1/(1-\gamma))}{1-\gamma}}$	[26, 20]
Policy Iteration (Block Simplex)	$\frac{ \mathcal{S} ^4 \mathcal{A} ^2}{1-\gamma} \log(\frac{1}{1-\gamma})$	[30],[25]
LP Algorithm	$(\mathcal{S} ^2 \mathcal{A} + \mathcal{S} ^{2.3729})\sqrt{ \mathcal{S} } \log \frac{1}{\epsilon}$	[19]
Combinatorial Interior Point Algorithm	$ \mathcal{S} ^4 \mathcal{A} ^4 \log \frac{ \mathcal{S} }{1-\gamma}$	[29]
Algorithm 1 for general DMDP	$\tilde{O}(\frac{ \mathcal{S} ^3 \mathcal{A} }{(1-\gamma)^4\epsilon^2})$	Corollary 1
Algorithm 1 for ergodic DMDP	$\tilde{O}(\frac{ \mathcal{S} \mathcal{A} }{(1-\gamma)^2\epsilon^2})$	Corollary 2

Table 1: Running Time Complexity for solving DMDP. In this table, $|\mathcal{S}|$ is the number of states, $|\mathcal{A}|$ is the number of actions per state, $\gamma \in (0, 1)$ is the discount factor, and L is the total bit size to present the DMDP input.

Our new method and analysis is motivated by the line of developments on stochastic first-order optimization methods. Stochastic first-order methods originate from the stochastic approximation method for the root finding problem; see [18, 2, 6]. They find wide applications in stochastic convex optimization, especially problems arising from machine learning applications. Several studies have been conducted to analyze the sample complexity of stochastic first-order methods, started by the seminal work [22] and followed by many others.

In particular, our proposed method is closely related to stochastic first-order method for saddle point problems. The earliest work of this type is by [23], which studied a stochastic approximation method for convex-concave saddle point problem without any convergence rate analysis. Two later works [11, 9] studied optimal rate of convergence for strongly convex-concave saddle point problems under a stochastic first-order oracle. Another related work is by [10], which developed a class of sublinear algorithms for minimax problems from machine learning. It showed that it is possible to find a pair of primal-dual solutions achieving ϵ duality gap in running time $\frac{1}{\epsilon^2}$. Unfortunately, none of these existing results on the stochastic saddle point problem applies to the DMDP. There are two prior attempts to use primal-iteration for online policy estimation of MDP. The work [27] considered the stochastic primal-dual iteration to solve DMDP without explicit complexity analysis. Later the work [8] proposed a more refined primal-dual method and established a sample complexity upper bound $\mathcal{O}(|\mathcal{S}|^{4.5}|\mathcal{A}|\epsilon^{-2})$. No running-time complexity analysis is available.

3 Bellman Equation, Linear Programs, and Stochastic Saddle Point Problem

Consider a DMDP tuple $\mathcal{M} = (\mathcal{S}, \mathcal{A}, \mathcal{P}, \mathbf{r}, \gamma)$. For a fixed policy π , the value vector $\mathbf{v}^\pi \in \mathbb{R}^{|\mathcal{S}|}$ is defined as

$$v_i^\pi = \mathbf{E}^\pi \left[\sum_{t=1}^{\infty} \gamma^t r_{i_t i_{t+1}}(a_t) \mid i_1 = i \right], \quad \forall i \in \mathcal{S},$$

where $\mathbf{E}^\pi[\cdot]$ is taken over the state-action trajectory $\{i_1, a_1, i_2, a_2, \dots\}$ generated by the Markov decision process under policy π . The optimal value vector $\mathbf{v}^* \in \mathbb{R}^{|\mathcal{S}|}$ is defined as

$$v_i^* = \max_{\pi} \mathbf{E}^\pi \left[\sum_{t=1}^{\infty} \gamma^t r_{i_t i_{t+1}}(a_t) \mid i_1 = i \right] = \mathbf{E}^{\pi^*} \left[\sum_{t=1}^{\infty} \gamma^t r_{i_t i_{t+1}}(a_t) \mid i_1 = i \right], \quad \forall i \in \mathcal{S}.$$

According to the theory of dynamic programming [24, 3], a vector \mathbf{v}^* is the optimal value function to the DMDP if and only if it satisfies the following $|\mathcal{S}| \times |\mathcal{S}|$ system of equations, known as the *Bellman equation*, given by

$$v_i^* = \max_{a \in \mathcal{A}} \left\{ \gamma \sum_{j \in \mathcal{S}} p_{ij}(a) v_j^* + \sum_{j \in \mathcal{S}} p_{ij}(a) r_{ij}(a) \right\}, \quad \forall i \in \mathcal{S},$$

When $\gamma \in (0, 1)$, the Bellman equation has a unique fixed point solution \mathbf{v}^* , and it equals to the optimal value vector of the DMDP. Moreover, a policy π^* is an optimal policy of the DMDP if and only if it attains the elementwise maximization in the Bellman equation. For finite-state DMDP, there always exists at least one optimal policy π^* . If the optimal policy is unique, it is also a deterministic policy. If there are multiple optimal policies, there exist infinitely many optimal randomized policies.

The nonlinear Bellman equation is equivalent to the following $|\mathcal{S}| \times (|\mathcal{S}| |\mathcal{A}|)$ linear programming problem (see [24] Section 6.9 and the paper [13]):

$$\begin{aligned} & \text{minimize } (1 - \gamma) \mathbf{q}^\top \mathbf{v} \\ & \text{subject to } (I - \gamma P_a) \mathbf{v} - \mathbf{r}_a \geq 0, \quad \forall a \in \mathcal{A}, \end{aligned} \quad (1)$$

where $\mathbf{q} \in \mathbb{R}^{|\mathcal{S}|}$ is an arbitrary vector of probability distribution satisfying $\mathbf{e}^\top \mathbf{q} = 1$ and $\mathbf{q} > 0$, $P_a \in \mathbb{R}^{|\mathcal{S}| \times |\mathcal{S}|}$ is matrix whose (i, j) -th entry equals to $p_{ij}(a)$, I is the identity matrix with dimension $|\mathcal{S}| \times |\mathcal{S}|$ and $\mathbf{r}_a \in \mathbb{R}^{|\mathcal{S}|}$ is the expected state-transition reward vector under action a , i.e., $r_a(i) = \sum_{j \in \mathcal{S}} p_{ij}(a) r_{ij}(a)$ for all $i \in \mathcal{S}$. The dual linear program of (1) is

$$\begin{aligned} & \text{maximize } \sum_{a \in \mathcal{A}} \mu_a^\top \mathbf{r}_a \\ & \text{subject to } \sum_{a \in \mathcal{A}} \left(I - \gamma P_a^\top \right) \mu_a = (1 - \gamma) \mathbf{q}, \quad \mu_a \geq 0, \quad \forall a \in \mathcal{A}. \end{aligned} \quad (2)$$

It is well known that each deterministic policy corresponds to a basic feasible solution to the dual linear program (2). A randomized policy is a mixture of deterministic policies, so it corresponds to a feasible solution of program (2). We denote by $\mu^* = (\mu_a^*)_{a \in \mathcal{A}}$ the optimal solution to the dual linear program (2). If there is a unique dual solution, it must be a basic feasible solution. In this case, it is well known that the basis of μ^* corresponds to an optimal deterministic policy.

We formulate the linear programs (1)-(2) into an equivalent minimax problem, given by

$$\min_{\mathbf{v} \in \mathcal{V}} \max_{\mu \in \mathcal{U}_{\theta, \mathbf{q}}} (1 - \gamma) \mathbf{q}^\top \mathbf{v} + \sum_{a \in \mathcal{A}} \mu_a^\top ((\gamma P_a - I) \mathbf{v} + \mathbf{r}_a). \quad (3)$$

We construct \mathcal{V} and $\mathcal{U}_{\theta, \mathbf{q}}$ to be the search spaces for the value and the policy, respectively, given by

$$\mathcal{V} = \left\{ \|\mathbf{v}\|_\infty \leq \frac{1}{1 - \gamma}, \mathbf{v} \geq 0 \right\}, \quad \mathcal{U}_{\theta, \mathbf{q}} = \left\{ \mathbf{e}^\top \mu = 1, \mu \geq 0, \sum_{a \in \mathcal{A}} \mu_a \geq \theta \mathbf{q} \right\},$$

where θ is a small value to be specified. Let us verify that $\mathbf{v}^* \in \mathcal{V}$ and $\mu^* \in \mathcal{U}_{\theta, \mathbf{q}}$. Since all rewards $r_{ij}(a)$ belong to $[1/2, 1]$, we can easily verify that $0 \leq v_i^* \leq \frac{1}{1-\gamma}$ for all i , therefore $\mathbf{v}^* \in \mathcal{V}$. By multiplying both sides of the dual constraint $\sum_{a \in \mathcal{A}} (I - \gamma P_a^\top) \mu_a^* = (1 - \gamma) \mathbf{q}$ with \mathbf{e}^\top , we also verify that $\mathbf{e}^\top \mu^* = 1$ because \mathbf{q} is a probability distribution. In subsequent analysis, we will specify choices of θ, \mathbf{q} so that $\sum_{a \in \mathcal{A}} \mu_a^* \geq \theta$ and $\mu^* \in \mathcal{U}_{\theta, \mathbf{q}}$. As long as $\mathbf{v}^* \in \mathcal{V}$ and $\mu^* \in \mathcal{U}_{\theta, \mathbf{q}}$, we would be able to tackle the DMDP by approximately solving the saddle point problem (3).

There are many ways to formulate the minimax problem (3) into a stochastic saddle point problem. A key observation is that the probability transition matrix P_a and the reward vector \mathbf{r}_a are naturally expectations of some random variables. For example, we can rewrite (3) as follows:

$$\min_{\mathbf{v} \in \mathcal{V}} \max_{\mu \in \mathcal{U}_{\theta, \mathbf{q}}} (1 - \gamma) \mathbf{q}^\top \mathbf{v} + \sum_{a \in \mathcal{A}} \mu_a^\top \left(\left(\gamma \sum_{i \in \mathcal{S}} \mathbf{E}_{j|i, a} [\mathbf{e}_i \mathbf{e}_j^\top] - I \right) \mathbf{v} + \sum_{i \in \mathcal{S}} \mathbf{E}_{j|i, a} [r_{ij}(a) \mathbf{e}_i] \right),$$

where the expectation $\mathbf{E}_{j|i, a}[\cdot]$ is taken over $j \sim p_{ij}(a)$ where i and a are fixed. The preceding formulation suggests that one can simulate transitions of the Markov decision process in order to draw samples of the Lagrangian function and its partial derivatives. In addition, we will show later that simulation of the Markov decision process is almost *free* in the sense that each transition can be simulated in $\tilde{\mathcal{O}}(1)$ arithmetic operations. This motivates the use of a stochastic primal-dual iteration for solving the DMDP.

4 Randomized Linear Programming Algorithm

Motivated by the saddle point formulation of Bellman's equation, we develop a randomized linear programming method to compute an approximately optimal policy. The method is given in Algorithm 1. It is essentially a coordinate descent iteration with adaptive sampling. In each iteration, only a few coordinates of the primal vector (value function) and dual vector (policy function) are updated.

A key component of Algorithm 1 is the use of adaptive sampling of state-action pairs (i, a) based on the current policy (in Step 8). Each iteration requires sampling from (Step 8) and updating (Step 11) two dynamically updated probability distributions ξ^t and π^t . To be more general, we consider the following sampling problem: We are given a stream of updates to a m -dimensional vector of nonnegative weights \mathbf{w} . We want to output a random coordinate $i = 1, \dots, m$ with probability $\frac{w_i}{\|\mathbf{w}\|_1}$. For the general sampling problem, it is well known that by using a binary tree-based scheme developed by [28] one can update the weight vector in $\mathcal{O}(\log m)$ time and generate a new random coordinate in $\mathcal{O}(\log m)$ time. As a result, one can update and sample from a dynamically updated distribution using $\tilde{\mathcal{O}}(1)$ arithmetic operations per update and $\tilde{\mathcal{O}}(1)$ arithmetic operations per sample. As a result, Step 8 and Step 11 can be realized using $\tilde{\mathcal{O}}(1)$ running time.

Now we analyze the running-time complexity of Algorithm 1. Step 5 preprocesses the input probabilities $p_{ij}(a)$ for all $i, j \in \mathcal{S}$ and $a \in \mathcal{A}$ to $|\mathcal{S}||\mathcal{A}|$ samplers, one for each state-action pair. Each of the sampler corresponds to a probability vector of $|\mathcal{S}|$ dimension. The preprocessing time for each m -dimensional vector is $\tilde{\mathcal{O}}(m)$. Therefore the total processing time is $\tilde{\mathcal{O}}(S^2 A)$. Step 8 can be implemented by drawing one sample from ξ^t and one sample from π^t . As long as ξ^t, π^t

¹The preprocessing step can be skipped if the input data are given in the format of sampleable data structures, e.g., sorted arrays [7], or binary trees [28].

²It is not necessarily to actually execute the normalization steps $\xi \leftarrow \xi / \|\xi\|_1$ and $\pi_i \leftarrow \pi_i / \|\pi_i\|_1$. To update a probability distribution, one can maintain and update an un-normalized weight vector by one coordinate. The update can be performed in $\tilde{\mathcal{O}}(1)$ arithmetic operations [28].

Algorithm 1 Randomized Primal-Dual Method for DMDPs

- 1: **Input:** DMDP tuple $\mathcal{M} = (\mathcal{S}, \mathcal{A}, \mathcal{P}, \mathbf{r}, \gamma)$, $\mathbf{q} = \frac{1}{S}\mathbf{e}$, $\theta = (1 - \gamma)$, T .
 - 2: Set $\mathbf{v} = 0 \in \mathbb{R}^{|\mathcal{S}|}$
 - 3: Set $\xi = \frac{1}{|\mathcal{S}|}\mathbf{e} \in \mathbb{R}^{|\mathcal{S}|}$, $\pi_i = \frac{1}{|\mathcal{A}|}\mathbf{e} \in \mathbb{R}^{|\mathcal{A}|}$ for all $i \in \mathcal{S}$
 - 4: Set $\beta = (1 - \gamma)\sqrt{\frac{\log(|\mathcal{S}||\mathcal{A}|+1)}{2|\mathcal{S}||\mathcal{A}|T}}$, $\alpha = \frac{|\mathcal{S}|}{2(1-\gamma)^2}\beta$, $M = \frac{1}{1-\gamma}$
 - 5: Preprocess the input probabilities $\mathcal{P} = \{p_{ij}(a)\}$ for sampling (Complexity $\tilde{\mathcal{O}}(S^2A)^1$)
 - 6: **for** $t = 1, 2, \dots, T$ **do**
 - 7: Sample i with probability $((1 - \theta)\xi_i + \theta q_i)$ (Complexity $\tilde{\mathcal{O}}(1)$)
 - 8: Sample a with probability $\pi_{i,a}$ (Complexity $\tilde{\mathcal{O}}(1)$)
 - 9: Conditioned on (i, a) , sample j with probability $p_{ij}(a)$ (Complexity $\tilde{\mathcal{O}}(1)$)
 - 10: Update the iterates by (Complexity $\tilde{\mathcal{O}}(1)$)
$$\Delta \leftarrow \beta \cdot \frac{\gamma v_j - v_i + r_{ij}(a) - M}{((1 - \theta)\xi_i + \theta q_i)\pi_{i,a}}$$

$$v_i \leftarrow \max \left\{ \min \left\{ v_i - \alpha \left(\frac{(1 - \gamma)q_i}{(1 - \theta)\xi_i + \theta q_i} - 1 \right), \quad \frac{1}{1 - \gamma} \right\}, 0 \right\}$$

$$v_j \leftarrow \max \left\{ \min \left\{ v_j - \alpha \gamma, \quad \frac{1}{1 - \gamma} \right\}, 0 \right\}$$
 - 11: Update the iterates by (Complexity $\tilde{\mathcal{O}}(1)^2$)
$$\xi_i \leftarrow \xi_i + \xi_i \pi_{i,a} (\exp \{\Delta\} - 1), \xi \leftarrow \xi / \|\xi\|_1$$

$$\pi_{i,a} \leftarrow \pi_{i,a} \cdot \exp \{\Delta\}, \pi_i \leftarrow \pi_i / \|\pi_i\|_1$$
 - 12: **end for**
 - 13: **Output:** Averaged dual iterate $\hat{\pi}_i = \frac{1}{T} \sum_{k=1}^T \pi_i^k$ for all $i \in \mathcal{S}$
-

are updated appropriately, drawing a sample from either of them takes running time $\tilde{\mathcal{O}}(1)$. Step 8 draws a sample from a previously initialized static distribution, which takes $\tilde{\mathcal{O}}(1)$ arithmetic operations. Step 9 updates two entries of the value vector \mathbf{v} and uses running time $\tilde{\mathcal{O}}(1)$. Step 11 makes dynamic update to one coordinate of a weight vector, in order to use it for sampling in the next iteration. It is not necessary to explicitly execute the normalization steps $\xi \leftarrow \xi / \|\xi\|_1$ and $\pi_i \leftarrow \pi_i / \|\pi_i\|_1$. As a result, Step 11 can be implemented using $\tilde{\mathcal{O}}(1)$ arithmetic operations. Step 13 requires taking average of past iterates $\{\pi_i^t\}_{t=1}^T$. This can be achieved by maintaining an additional variable to record the running multiplicative weights and running average (e.g., using a special binary tree structure). The additional storage overhead is $\mathcal{O}(|\mathcal{S}||\mathcal{A}|)$, and the additional computation overhead is $\tilde{\mathcal{O}}(1)$ per iteration.

To sum up, the total running-time complexity of Algorithm 1 consists of two parts: complexity of preprocessing and complexity per iteration. Preprocessing takes $\tilde{\mathcal{O}}(S^2A)$ running time, which can be skipped if the input data are given in some data structure that enables immediate sampling (e.g., sorted arrays [7] or binary trees [28]). Each iteration of Algorithm 1 takes $\tilde{\mathcal{O}}(1)$ arithmetic operations. The space complexity of Algorithm 1 is $\mathcal{O}(|\mathcal{S}||\mathcal{A}|)$.

5 Complexity Analysis

In this section, we analyze the complexity of Algorithm 1. We observe that each iteration of Algorithm 1 essentially performs a primal-dual update. So let us analyze the convergence of the duality gap sequence. Our first result gives an upper bound for the expected duality gap after a finite number of iterations.

Theorem 1 (Duality Gap Bound). *Let $\mathcal{M} = (\mathcal{S}, \mathcal{A}, \mathcal{P}, \mathbf{r}, \gamma)$ be an arbitrary DMDP tuple, let $\mathbf{q} \in \mathbb{R}^{|\mathcal{S}|}$ be an arbitrary probability vector and let $\theta \geq 1 - \gamma$. Assume that the solution of the dual linear program (2) satisfies $\mu^* \in \mathcal{U}_{\theta, \mathbf{q}}$. Let Algorithm 1 iterate with the input $(\mathcal{M}, \mathbf{q}, \theta)$, then the sequence of iterates $\{(\xi^t, \pi^t)\}_{t=1}^T$ satisfies*

$$\mathbf{E} \left[\sum_{a \in \mathcal{A}, i \in \mathcal{S}} \hat{\mu}_{a,i} \left(v_i^* - \gamma \sum_{j \in \mathcal{S}} p_{ij}(a) v^*(j) - \sum_{j \in \mathcal{S}} p_{ij}(a) r_{ij}(a) \right) \right] \leq \frac{\sqrt{2|\mathcal{S}|(|\mathcal{A}| + 1) (\log(|\mathcal{S}||\mathcal{A}|) + 1)}}{(1 - \gamma)\sqrt{T}},$$

where $\hat{\mu}_{i,a} = \frac{1}{T} \sum_{t=1}^T ((1 - \theta)\xi_i^t + \theta q_i) \pi_{i,a}^t$.

Outline of Analysis. Algorithm 1 can be interpreted as a stochastic primal-dual iteration that applies to the saddle point problem (3). To see this, we define the auxiliary iterate $\mu_{i,a}^t$ as

$$\mu_{i,a}^t = ((1 - \theta)\xi_i^t + \theta q_i) \pi_{i,a}^t, \quad \forall i \in \mathcal{S}, a \in \mathcal{A}.$$

The t -th iteration of Algorithm 1 takes the form

$$\begin{aligned} \mathbf{v}^{t+1} &= \operatorname{argmin}_{\mathbf{v} \in \mathcal{V}} \left\{ (\partial_{\mathbf{v}} L(\mathbf{v}, \mu^t) + \varepsilon^t)^\top (\mathbf{v} - \mathbf{v}^t) + \frac{1}{2\beta} \|\mathbf{v} - \mathbf{v}^t\|^2 \right\}, \\ \mu^{t+1} &= \operatorname{argmin}_{\mu \in \mathcal{U}_{\theta, \mathbf{q}}} \left\{ (\partial_{\mu} L(\mathbf{v}^t, \mu^t) + \omega^t)^\top (\mu - \mu^t) + \frac{1}{2\alpha} \Phi(\mu; \mu^t) \right\}, \end{aligned}$$

where ε^t and ω^t are zero-mean random noises due to the sampling step, Φ is a Bregman divergence function given by

$$\Phi(\mu; \hat{\mu}) = (1 - \theta) D_{KL}(\lambda || \hat{\lambda}) + \theta \sum_{i \in \mathcal{S}} q_i D_{KL}(\pi_i || \hat{\pi}_i),$$

where (λ, π) is determined by μ and $(\hat{\lambda}, \hat{\pi})$ is determined by $\hat{\mu}$. The dual feasible region $\mathcal{U}_{\theta, \mathbf{q}}$ plays a role of the “information set,” in which we search for the optimal policy. The information set $\mathcal{U}_{\theta, \mathbf{q}}$ shall be constructed to characterize properties and additional prior knowledge (if there is any) regarding the DMDP. Clearly, the set $\mathcal{U}_{\theta, \mathbf{q}}$ and the divergence function Φ are determined by the input parameters θ, \mathbf{q} . We will specify values of the parameters θ, \mathbf{q} in subsequent analysis. We defer the detailed proof of Theorem 1 to Appendix A-B.

Now we present our first main result on the iteration complexity of Algorithm 1 for finding an ϵ -optimal policy.

Theorem 2 (Complexity Result for Arbitrary DMDP). *Let $\mathcal{M} = (\mathcal{S}, \mathcal{A}, \mathcal{P}, \mathbf{r}, \gamma)$ be an arbitrary DMDP, and let Algorithm 1 iterate with $\mathbf{q} = \frac{1}{|\mathcal{S}|} \mathbf{e}, \theta = (1 - \gamma)$ for the following iteration number*

$$T = \Omega \left(\frac{|\mathcal{S}|^3 |\mathcal{A}| \log(|\mathcal{S}||\mathcal{A}|)}{(1 - \gamma)^4 \epsilon^2} \right).$$

Then the output policy $\hat{\pi}$ satisfies $\mathbf{v}^{\hat{\pi}} \geq (1 - \epsilon) \mathbf{v}^$ with probability at least $2/3$.*

Theorem 2 establishes a worst-case iteration complexity for Algorithm 1. The result of Theorem 2 applies to all instances of DMDPs with $|\mathcal{S}|$ states, $|\mathcal{A}|$ actions per state, and a fixed discount factor γ . Theorem 2 does not require any additional property regarding the Markov decision process. It does not require irreducibility or aperiodicity of the associated Markov chains. In fact, Theorem 2 holds for instances of DMDP with transient states. It also holds for instances with multiple optimal policies. Its proof is deferred to Appendix B.

Furthermore, one can obtain a near-optimal policy with probability arbitrarily close to 1 by running Algorithm 1 for multiple independent trials. This would also require us to evaluate multiple candidate policies and select the best one out of many. In fact, we show that it is possible to approximately evaluate any policy π within ϵ precision in running time $\tilde{\mathcal{O}}(\frac{1}{\epsilon^2(1-\gamma)^2})$ for a fixed initial distribution (see Lemma 5 in Appendix C). Then we obtain the following corollary.

Corollary 1. *For any $\epsilon \in (0, 1)$, $\delta \in (0, 1)$ and initial distribution vector $\mathbf{q} \in \mathbb{R}^{|\mathcal{S}|}$, there exists an algorithm that outputs an approximately optimal policy $\hat{\pi}$ such that $\mathbf{q}^\top \mathbf{v}^{\hat{\pi}} \geq (1 - \epsilon)\mathbf{q}^\top \mathbf{v}^*$ in running time*

$$\tilde{\mathcal{O}} \left(|\mathcal{S}|^2 |\mathcal{A}| + \frac{|\mathcal{S}|^3 |\mathcal{A}|}{(1 - \gamma)^4 \epsilon^2} \log \left(\frac{1}{\delta} \right) + \frac{1}{\epsilon^2} \left(\log \frac{1}{\delta} \right)^2 \right)$$

with probability at least $1 - \delta$.

The proof of Corollary 1 is given in Appendix C. Comparing the preceding complexity result with the input size $\mathcal{O}(|\mathcal{S}|^2 |\mathcal{A}|)$ of the DMDP, we conclude that Algorithm 1 has nearly-linear running-time complexity in the worst case. For large-scale problems, as long as $|\mathcal{S}| \gg \frac{1}{\epsilon}$ and $|\mathcal{A}| \gg \frac{1}{\epsilon}$, Algorithm 1 is able to compute an approximate policy more efficiently than any known method.

However, the cubic dependence $|\mathcal{S}|^3$ in Theorem 2 and Corollary 1 is not quite satisfying. We conjecture that it can be improved to $|\mathcal{S}|^2$ (or even $|\mathcal{S}|$) using an improved algorithm and analysis. In addition, we conjecture that the complexity result should have a better dependence on $\frac{1}{1-\gamma}$, especially when all P^π 's have relatively large spectral gaps. These two questions are left open for future investigation.

Next we will see that, the DMDP becomes easier to solve when the associated Markov process is “better” behaved. In what follows, we focus on the class of DMDPs where every stationary policy generates an ergodic Markov chain. For an arbitrary policy π , we define ν^π to be the *stationary distribution under policy π* , i.e., $\nu^\pi = (P^\pi)^\top \nu^\pi$. Our next main result shows that Algorithm 1 has significantly improved complexity guarantee for ergodic DMDP.

Theorem 3 (Complexity Result for Ergodic DMDP). *Let $\mathcal{M} = (\mathcal{S}, \mathcal{A}, \mathcal{P}, \mathbf{r}, \gamma)$ be an DMDP tuple. Assume that there exists $c_2 \geq c_1 > 0$ and a distribution vector $\mathbf{q} \in \mathbb{R}^{|\mathcal{S}|}$ such that the Markov decision process is ergodic under any policy and satisfies*

$$c_1 \mathbf{q} \leq \nu^\pi \leq c_2 \mathbf{q},$$

for any stationary policy π . Let the input be \mathcal{M}, \mathbf{q} , $\theta = 1 - \gamma + \gamma(\frac{c_1}{c_2})$, and let Algorithm 1 iterate with the following iteration number

$$T = \Omega \left(\left(\frac{c_2}{c_1} \right)^4 \frac{|\mathcal{S}| |\mathcal{A}| \log(|\mathcal{S}| |\mathcal{A}|)}{(1 - \gamma)^2 \epsilon^2} \right),$$

then the output policy $\hat{\pi}$ satisfies $\mathbf{q}^\top \mathbf{v}^{\hat{\pi}} \geq (1 - \epsilon)\mathbf{q}^\top \mathbf{v}^$ with probability at least $2/3$.*

Theorem 3 shows that the iteration complexity of Algorithm 1 substantially improves when the DMDP is ergodic under every stationary policy. More specifically, the complexity reduces when all policies generate “similar” stationary distributions. In the next corollary, we establish the total running-time complexity for obtaining a near-optimal policy with high probability.

Corollary 2. *Let the assumptions of Theorem 3 hold. For any $\epsilon \in (0, 1)$, $\delta \in (0, 1)$, there exists an algorithm that outputs an approximately optimal policy $\hat{\pi}$ such that $\mathbf{q}^\top \mathbf{v}^{\hat{\pi}} \geq (1 - \epsilon)\mathbf{q}^\top \mathbf{v}^*$ in running time*

$$\tilde{O} \left(|\mathcal{S}|^2 |\mathcal{A}| + \left(\frac{c_2}{c_1} \right)^4 \frac{|\mathcal{S}| |\mathcal{A}|}{(1 - \gamma)^2 \epsilon^2} \log \left(\frac{1}{\delta} \right) + \frac{1}{\epsilon^2} \left(\log \frac{1}{\delta} \right)^2 \right)$$

with probability at least $1 - \delta$.

In the result of Corollary 2, the first term $|\mathcal{S}|^2 |\mathcal{A}|$ is due to Algorithm 1’s initialization step 5, which preprocesses the input probabilities into samplers of state transitions. When the input data are given in a way that enables immediate sampling, the total running-time complexity reduces to

$$\tilde{O} \left(\left(\frac{c_2}{c_1} \right)^4 \frac{|\mathcal{S}| |\mathcal{A}|}{(1 - \gamma)^2 \epsilon^2} \right)$$

(ignoring polylog factors of $\frac{1}{\delta}$). The preceding running-time complexity is almost linear in $|\mathcal{S} \times \mathcal{A}|$. It means that each state-action pair is queried for a constant number of times on average. This result is sublinear with respect to the input size $\mathcal{O}(|\mathcal{S}|^2 |\mathcal{A}|)$. It suggests that one can find an approximately optimal policy without even reading all the input entries. We recall that the input data mainly consist of transition probabilities $p_{ij}(a)$. An explanation for the sublinear complexity is that certain small probabilities in the input data can be safely ignored, without deteriorating the quality of the approximate policy significantly.

The ratio $\frac{c_2}{c_1}$ characterizes a notion of complexity of ergodic DMDP. Intuitively, dynamic programs are harder to solve when different policies lead to vastly different state trajectories. For example, suppose that there is a critical state that can only show up after a particular sequence of correct actions (this typically happens in aperiodic Markov chains). In this case, any algorithm would need to search over the space of all policies to identify the critical state. For another example, consider that all policies only affect the immediate reward but will lead to the same outgoing transition probabilities. In this case, one would be able to learn the optimal action at each state much more efficiently, where $c_1/c_2 = 1$.

We remark that Theorem 3 requires the ratio $\frac{c_1}{c_2}$ as part of the input to Algorithm 1. This is a weakness because it requires prior knowledge about the DMDP. We conjecture that this requirement can be relaxed by using an improved analysis. This question is left to be answered in future research.

6 Remarks

We have developed a novel randomized method that exploits the linear duality between the value function and the policy function for solving DMDPs. It is related to several fundamental methods in linear programming, stochastic optimization and online learning. It is easy to implement, uses sublinear space complexity and nearly-linear (sometimes sublinear) running-time complexity.

Algorithm 1 can be viewed as a randomized version of the simplex method, which is also equivalent to a version of the policy iteration method for solving MDP. It maintains a primal variable (value) and a dual variable (policy). Each update $\pi_{i,a} \leftarrow \pi_{i,a} \cdot \exp \{ \Delta \}$ mimics a pivoting step towards a neighboring basis. Instead of moving from one basis to another one, each update in

the dual variable can be viewed as a “soft” pivot. Our theoretical results show that the algorithm finds an approximate policy in $\tilde{O}(SA)$ iterations. This is somewhat similar to the simplex method, which also terminates in $\tilde{O}(SA)$ (which is the number of constraints) iterations on average. What makes our randomized algorithm different is its $\tilde{O}(1)$ running time per iteration. It avoids explicitly solving any linear system, which is unavoidable in the simplex method.

Algorithm 1 can also be viewed as a stochastic approximation method for solving a saddle point problem. It utilizes the structures of specially crafted primal and dual constraint sets to make each update as simple and efficient as possible. The update of the dual variable (policy) uses a special Bregman divergence function which is related to the relative entropy between randomized policies.

It is worth noting that Algorithm 1 is related to the exponentiated gradient method for the multi-arm bandit problem in the online learning setting. When there is a single state, Algorithm 1 reduces to the basic exponentiated gradient method. This observation provides a hint that we might be able to adapt Algorithm 1 to apply to online reinforcement learning. This is a direction for future research.

The new method and complexity results of this paper suggest a promising direction that awaits further research. The current results leave open many questions. We conjecture that the complexity result should have a better dependence on $\frac{1}{1-\gamma}$, especially when all P^π ’s have relatively large spectral gaps. A related notion of complexity metric for MDP is the diameter, i.e., the maximal expected time to move from any state to any other state. We conjecture that the diameter should play a key role in an improved complexity analysis and replace at least one factor of $\frac{1}{1-\gamma}$. We also conjecture that the sublinear-time complexity result of Theorem 3 hold for more general DMDPs without prior knowledge about $\frac{c_2}{c_1}$. Another direction for future research is to consider the running-time complexity for finite-horizon MDP and average-reward MDP. It remains unclear what roles the mixing rate and the horizon play in the running-time complexity. In the mean time, an equally important (if not more) question is to establish the computation complexity lower bound for approximating optimal policies of MDP.

References

- [1] Richard Bellman. *Dynamic Programming*. Princeton University Press, Princeton, NJ, 1957.
- [2] Albert Benveniste, Michel Métivier, and Pierre Priouret. *Adaptive algorithms and stochastic approximations*, volume 22. Springer Science & Business Media, 2012.
- [3] Dimitri P Bertsekas. *Dynamic programming and optimal control*, volume 1. Athena Scientific, Belmont, MA, 1995.
- [4] Dimitri P Bertsekas. *Abstract dynamic programming*. Athena Scientific, Belmont, MA, 2013.
- [5] Dimitri P Bertsekas and John N Tsitsiklis. Neuro-dynamic programming: an overview. In *Proceedings of the 34th IEEE Conference on Decision and Control*, volume 1, pages 560–564. IEEE, 1995.
- [6] Vivek S. Borkar. *Stochastic approximation: a dynamical systems viewpoint*. Cambridge University Press, Cambridge, 2008.
- [7] Karl Bringmann and Konstantinos Panagiotou. Efficient sampling methods for discrete distributions. In *International Colloquium on Automata, Languages, and Programming*, pages 133–144. Springer, 2012.
- [8] Yichen Chen and Mengdi Wang. Stochastic primal-dual methods and sample complexity of reinforcement learning. *arXiv preprint arXiv:1612.02516*, 2016.
- [9] Yunmei Chen, Guanghui Lan, and Yuyuan Ouyang. Optimal primal-dual methods for a class of saddle point problems. *SIAM Journal on Optimization*, 24(4):1779–1814, 2014.

- [10] Kenneth L Clarkson, Elad Hazan, and David P Woodruff. Sublinear optimization for machine learning. *Journal of the ACM (JACM)*, 59(5):23, 2012.
- [11] Cong Dang and Guanghui Lan. Randomized first-order methods for saddle point optimization. *arXiv preprint arXiv:1409.8625*, 2014.
- [12] George Dantzig. *Linear Programming and Extensions*. Princeton University Press, Princeton, NJ, 2016.
- [13] Daniela Pucci de Farias and Benjamin Van Roy. The linear programming approach to approximate dynamic programming. *Operations Research*, 51(6):850–865, 2003.
- [14] Guy De Ghellinck. Les problemes de decisions sequentielles. *Cahiers du Centre dEtudes de Recherche Opérationnelle*, 2(2):161–179, 1960.
- [15] F d’Epenoux. A probabilistic production and inventory problem. *Management Science*, 10(1):98–108, 1963.
- [16] Eugene A Feinberg and Jefferson Huang. The value iteration algorithm is not strongly polynomial for discounted dynamic programming. *Operations Research Letters*, 42(2):130–131, 2014.
- [17] Ronald A. Howard. *Dynamic programming and Markov processes*. The MIT press, Cambridge, MA, 1960.
- [18] Harold J Kushner and George Yin. *Stochastic Approximation and Recursive Algorithms and Applications*. Springer, 2003.
- [19] Yin Tat Lee and Aaron Sidford. Path finding methods for linear programming: solving linear programs in $O(\sqrt{\text{rank}})$ iterations and faster algorithms for maximum flow. In *2014 IEEE 55th Annual Symposium on Foundations of Computer Science*, pages 424–433, Oct 2014.
- [20] Michael L Littman, Thomas L Dean, and Leslie Pack Kaelbling. On the complexity of solving Markov decision problems. In *Proceedings of the Eleventh conference on Uncertainty in artificial intelligence*, pages 394–402. Morgan Kaufmann Publishers Inc., 1995.
- [21] Yishay Mansour and Satinder Singh. On the complexity of policy iteration. In *Proceedings of the Fifteenth conference on Uncertainty in artificial intelligence*, pages 401–408. Morgan Kaufmann Publishers Inc., 1999.
- [22] Arkadi Nemirovski, Anatoli Juditsky, Guanghui Lan, and Alexander Shapiro. Robust stochastic approximation approach to stochastic programming. *SIAM Journal on optimization*, 19(4):1574–1609, 2009.
- [23] Arkadi Nemirovski and Reuven Y Rubinstein. An efficient stochastic approximation algorithm for stochastic saddle point problems. In *Modeling Uncertainty*, pages 156–184. Springer, 2005.
- [24] Martin L Puterman. *Markov decision processes: discrete stochastic dynamic programming*. John Wiley & Sons, 2014.
- [25] Bruno Scherrer. Improved and generalized upper bounds on the complexity of policy iteration. In *Advances in Neural Information Processing Systems*, pages 386–394, 2013.
- [26] Paul Tseng. Solving h-horizon, stationary markov decision problems in time proportional to $\log(h)$. *Operations Research Letters*, 9(5):287–297, 1990.
- [27] Mengdi Wang and Yichen Chen. An online primal-dual method for discounted Markov decision processes. In *IEEE Conference of Decisions and Control*, 2016.
- [28] Chak-Kuen Wong and Malcolm C. Easton. An efficient method for weighted sampling without replacement. *SIAM Journal on Computing*, 9(1):111–113, 1980.
- [29] Yinyu Ye. A new complexity result on solving the Markov decision problem. *Mathematics of Operations Research*, 30(3):733–749, 2005.
- [30] Yinyu Ye. The simplex and policy-iteration methods are strongly polynomial for the Markov decision problem with a fixed discount rate. *Mathematics of Operations Research*, 36(4):593–603, 2011.

A Technical Lemmas

In this section, we analyze the convergence of Algorithm 1. We denote the t -th iterates generated by Algorithm 1 by π^t , ξ^t , and \mathbf{v}^t . We define the auxiliary variables $\lambda = \left(\lambda_{i,a}^t\right)_{i \in \mathcal{S}, a \in \mathcal{A}}$ as

$$\lambda_{i,a}^t = \xi_i^t \pi_{i,a}^t.$$

According to Algorithm 1, we can verify that $\xi^t \in \mathbb{R}^{|\mathcal{S}|}$ and $\pi_i^t \in \mathbb{R}^{|\mathcal{A}|}$ are vectors of probability distributions. It follows that λ^t is always a $|\mathcal{S}||\mathcal{A}|$ -dimensional vector of probability distribution. In addition, the updates on ξ^t and π^t can be equivalently written as updates on λ^t and π^t , given by

$$\lambda_{i,a}^{t+1} = \frac{\lambda_{i,a}^t \cdot \exp(\Delta_{i,a}^{t+1})}{\sum_{i',a'} \lambda_{i',a'}^t \cdot \exp(\Delta_{i',a'}^{t+1})}, \quad \pi_{i,a}^{t+1} = \frac{\pi_{i,a}^t \cdot \exp(\Delta_{i,a}^{t+1})}{\sum_{a'} \pi_{i,a'}^t \cdot \exp(\Delta_{i,a'}^{t+1})} \quad \forall i \in \mathcal{S}, a \in \mathcal{A}, \quad (4)$$

where

$$\Delta_{i,a}^{t+1} = \begin{cases} \beta \cdot \frac{(\gamma v_j^t - v_i^t + r_{ijt}(a) - M)}{((1-\theta)\xi_i^t + \theta q_i)\pi_{i,a}^t} & \text{if } i = i_{t+1}, a = a_{t+1} \\ 0 & \text{otherwise} \end{cases} \quad (5)$$

In what follows, we denote by \mathcal{F}_t the collection of random variables that are revealed up to the end of the t -th iteration. We denote by $\mu_{i,a}^t$ the dual iterates given by

$$\mu_{i,a}^t = ((1-\theta)\xi_i^t + \theta q_i)\pi_{i,a}^t = (1-\theta)\lambda_{i,a}^t + \theta q_i \pi_{i,a}^t.$$

According to Algorithm 1, we can verify that $\mu^t \in \mathcal{U}_{\theta, \mathbf{q}}$ and $\mathbf{v}^t \in \mathcal{V}$ for all t with probability 1.

Lemma 1. *If $\mu^* \in \mathcal{U}_{\theta, \mathbf{q}}$, there exists probability distribution vectors $\lambda^* \in \mathbb{R}^{|\mathcal{S}||\mathcal{A}|}$ and $\pi_i^* \in \mathbb{R}^{|\mathcal{A}|}$, $i \in \mathcal{S}$, such that*

$$\mu_{i,a}^* = (1-\theta)\lambda_{i,a}^* + \theta q_i \pi_{i,a}^*, \quad \forall i \in \mathcal{S}, a \in \mathcal{A}.$$

Proof. The proof is straightforward by the definition of $\mathcal{U}_{\theta, \mathbf{q}}$. ■

Lemma 2. *The iterates generated by Algorithm 1 satisfy*

$$\mathbf{E} [D_{KL}(\lambda^* || \lambda^{t+1}) | \mathcal{F}_t] - D_{KL}(\lambda^* || \lambda^t) \leq \sum_{i \in \mathcal{S}} \sum_{a \in \mathcal{A}} (\lambda_{i,a}^t - \lambda_{i,a}^*) \mathbf{E} [\Delta_{i,a}^{t+1} | \mathcal{F}_t] + \frac{1}{2} \sum_{i \in \mathcal{S}} \sum_{a \in \mathcal{A}} \lambda_{i,a}^t \mathbf{E} \left[\left(\Delta_{i,a}^{t+1} \right)^2 | \mathcal{F}_t \right], \quad (6)$$

for all t , with probability 1.

Proof. By using the relation (4), we have

$$\begin{aligned} D_{KL}(\lambda^* || \lambda^{t+1}) - D_{KL}(\lambda^* || \lambda^t) &= \sum_{i \in \mathcal{S}} \sum_{a \in \mathcal{A}} \lambda_{i,a}^* \log \frac{\lambda_{i,a}^*}{\lambda_{i,a}^{t+1}} - \sum_{i \in \mathcal{S}} \sum_{a \in \mathcal{A}} \lambda_{i,a}^* \log \frac{\lambda_{i,a}^*}{\lambda_{i,a}^t} \\ &= \sum_{i \in \mathcal{S}} \sum_{a \in \mathcal{A}} \lambda_{i,a}^* \log \frac{\lambda_{i,a}^t}{\lambda_{i,a}^{t+1}} \\ &= \sum_{i \in \mathcal{S}} \sum_{a \in \mathcal{A}} \lambda_{i,a}^* \log \frac{Z}{\exp(\Delta_{i,a}^{t+1})} \\ &= \sum_{i \in \mathcal{S}} \sum_{a \in \mathcal{A}} \lambda_{i,a}^* \log(Z) - \sum_{i \in \mathcal{S}} \sum_{a \in \mathcal{A}} \lambda_{i,a}^* \Delta_{i,a}^{t+1} \\ &= \log Z - \sum_{i \in \mathcal{S}} \sum_{a \in \mathcal{A}} \lambda_{i,a}^* \Delta_{i,a}^{t+1}, \end{aligned} \quad (7)$$

where $Z = \sum_{i \in \mathcal{S}} \sum_{a \in \mathcal{A}} \lambda_{i,a}^t \exp(\Delta_{i,a}^{t+1})$. According to (5), we have $\gamma v_j^t - v_i^t + r_{ij}^t(a) - M \leq \frac{\gamma}{1-\gamma} - 0 + 1 - \frac{1}{1-\gamma} \leq 0$ because $v_i \in [0, \frac{1}{1-\gamma}]$, $r_{ij}^t(a) \in [\frac{1}{2}, 1]$ and $M = \frac{1}{1-\gamma}$. It follows that $\Delta_{i,a}^{t+1} \leq 0$ for all $i \in \mathcal{S}, a \in \mathcal{A}$ with probability 1. Then we derive

$$\begin{aligned} \log Z &= \log \left(\sum_{i \in \mathcal{S}} \sum_{a \in \mathcal{A}} \lambda_{i,a}^t \exp(\Delta_{i,a}^{t+1}) \right) \leq \log \sum_{i \in \mathcal{S}} \sum_{a \in \mathcal{A}} \lambda_{i,a}^t \left(1 + \Delta_{i,a}^{t+1} + \frac{1}{2} (\Delta_{i,a}^{t+1})^2 \right) \\ &= \log \left(1 + \sum_{i \in \mathcal{S}} \sum_{a \in \mathcal{A}} \lambda_{i,a}^t \Delta_{i,a}^{t+1} + \frac{1}{2} \sum_{i \in \mathcal{S}} \sum_{a \in \mathcal{A}} \lambda_{i,a}^t (\Delta_{i,a}^{t+1})^2 \right) \quad (8) \\ &\leq \sum_{i \in \mathcal{S}} \sum_{a \in \mathcal{A}} \lambda_{i,a}^t \Delta_{i,a}^{t+1} + \frac{1}{2} \sum_{i \in \mathcal{S}} \sum_{a \in \mathcal{A}} \lambda_{i,a}^t (\Delta_{i,a}^{t+1})^2, \end{aligned}$$

where the first inequality uses the fact $e^x \leq 1 + x + \frac{1}{2}x^2$ if $x \leq 0$ and the second inequality uses the fact $\log(1+x) \leq x$ for all x . We combine (7) and (8) and take conditional expectation $\mathbf{E}[\cdot | \mathcal{F}_t]$ on both sides, then we obtain (6). \blacksquare

Lemma 3. *For any $i \in \mathcal{S}$, the iterates generated by Algorithm 1 satisfy*

$$\mathbf{E} \left[D_{KL}(\pi_i^* || \pi_i^{t+1}) \mid \mathcal{F}_t \right] - D_{KL}(\pi_i^* || \pi_i^t) \leq \sum_{a \in \mathcal{A}} (\pi_{i,a}^t - \pi_{i,a}^*) \mathbf{E} \left[\Delta_{i,a}^{t+1} \mid \mathcal{F}_t \right] + \frac{1}{2} \sum_{a \in \mathcal{A}} \pi_{i,a}^t \mathbf{E} \left[(\Delta_{i,a}^{t+1})^2 \mid \mathcal{F}_t \right], \quad (9)$$

for all $t \geq 1$, with probability 1.

Proof. The proof is similar to Lemma 2. We omit it for simplicity. \blacksquare

Lemma 4. *The iterates generated by Algorithm 1 satisfy*

$$\sum_{i \in \mathcal{S}} \sum_{a \in \mathcal{A}} \mu_{i,a}^t \mathbf{E} \left[(\Delta_{i,a}^{t+1})^2 \mid \mathcal{F}_t \right] \leq \frac{4|\mathcal{S}||\mathcal{A}|\beta^2}{(1-\gamma)^2},$$

for all $t \geq 1$ with probability 1.

Proof. We note that $\mathbf{P}(i_{t+1} = i, a_{t+1} = a \mid \mathcal{F}_t) = \mu_{i,a}^t$. Then we have

$$\begin{aligned} \sum_{i \in \mathcal{S}} \sum_{a \in \mathcal{A}} \mu_{i,a}^t \mathbf{E} \left[(\Delta_{i,a}^{t+1})^2 \mid \mathcal{F}_t \right] &= \sum_{i \in \mathcal{S}} \sum_{a \in \mathcal{A}} \mu_{i,a}^t \cdot \mu_{i,a}^t \cdot \sum_{j \in \mathcal{S}} p_{ij}(a) \left(\beta \cdot \frac{(\gamma v_j^t - v_i^t + r_{ij}(a) - M)}{\mu_{i,a}^t} \right)^2 \\ &= \sum_{i \in \mathcal{S}} \sum_{a \in \mathcal{A}} \sum_{j \in \mathcal{S}} p_{ij}(a) (\beta \cdot (\gamma v_j^t - v_i^t + r_{ij}(a) - M))^2 \\ &\leq \sum_{i \in \mathcal{S}} \sum_{a \in \mathcal{A}} \left(\beta \cdot \frac{2}{1-\gamma} \right)^2 \\ &= \frac{4|\mathcal{S}||\mathcal{A}|\beta^2}{(1-\gamma)^2}, \end{aligned}$$

where the inequality uses the fact that \mathbf{v}^t belongs to the $\|\cdot\|_\infty$ ball of radius $\frac{1}{1-\gamma}$ and $M = \frac{1}{1-\gamma}$. \blacksquare

Proposition 1. We let $\Phi(\mu^t)$ be the divergence function given by

$$\Phi(\mu^t) = (1 - \theta)D_{KL}(\lambda^* || \lambda^t) + \theta \sum_{i \in \mathcal{S}} q_i D_{KL}(\pi_i^* || \pi_i^t).$$

The iterates generated by Algorithm 1 satisfy

$$\mathbf{E} [\Phi(\mu^{t+1}) | \mathcal{F}_t] \leq \Phi(\mu^t) + \beta \sum_{a \in \mathcal{A}} (\mu_a^t - \mu_a^*)^\top ((\gamma P_a - I)\mathbf{v}^t + \mathbf{r}_a) + \frac{2|\mathcal{S}||\mathcal{A}|\beta^2}{(1 - \gamma)^2}, \quad (10)$$

for all t , with probability 1.

Proof. We take the weighted sum between (6) and (9), so we have

$$\begin{aligned} \mathbf{E} [\Phi(\mu^{t+1}) | \mathcal{F}_t] &\leq \Phi(\mu^t) + (1 - \theta) \sum_{i \in \mathcal{S}} \sum_{a \in \mathcal{A}} (\lambda_{i,a}^t - \lambda_{i,a}^*) \mathbf{E} [\Delta_{i,a}^{t+1} | \mathcal{F}_t] + \frac{1 - \theta}{2} \sum_{i \in \mathcal{S}} \sum_{a \in \mathcal{A}} \lambda_{i,a}^t \mathbf{E} \left[\left(\Delta_{i,a}^{t+1} \right)^2 | \mathcal{F}_t \right] \\ &\quad + \theta \sum_{i \in \mathcal{S}} q_i \sum_{a \in \mathcal{A}} (\pi_{i,a}^t - \pi_{i,a}^*) \mathbf{E} [\Delta_{i,a}^{t+1} | \mathcal{F}_t] + \frac{\theta}{2} \sum_{i \in \mathcal{S}} q_i \sum_{a \in \mathcal{A}} \pi_{i,a}^t \mathbf{E} \left[\left(\Delta_{i,a}^{t+1} \right)^2 | \mathcal{F}_t \right] \\ &= \Phi(\mu^t) + \sum_{i \in \mathcal{S}} \sum_{a \in \mathcal{A}} (\mu_{i,a}^t - \mu_{i,a}^*) \mathbf{E} [\Delta_{i,a}^{t+1} | \mathcal{F}_t] + \frac{1}{2} \sum_{i \in \mathcal{S}} \sum_{a \in \mathcal{A}} \mu_{i,a}^t \mathbf{E} \left[\left(\Delta_{i,a}^{t+1} \right)^2 | \mathcal{F}_t \right], \end{aligned}$$

where the equality uses the relations $\mu_{i,a}^t = (1 - \theta)\lambda_{i,a}^t + \theta q_i \pi_{i,a}^t$ and $\mu_{i,a}^* = (1 - \theta)\lambda_{i,a}^* + \theta q_i \pi_{i,a}^*$ (by Lemma 1 since $\mu^* \in \mathcal{U}_{\theta, \mathbf{q}}$). For arbitrary $i \in \mathcal{S}$ and $a \in \mathcal{A}$, we have

$$\frac{1}{\beta} \cdot \mathbf{E} [\Delta_{i,a}^{t+1} | \mathcal{F}_t] = \gamma \sum_{j \in \mathcal{S}} p_{ij}(a) v_j^t - v_i^t + \sum_{j \in \mathcal{S}} p_{ij}(a) r_{ij}(a) - M = (\gamma P_a \mathbf{v}^t - \mathbf{v}^t + \mathbf{r}_a)_i - M.$$

It follows that

$$\begin{aligned} \frac{1}{\beta} \cdot \sum_{i \in \mathcal{S}} \sum_{a \in \mathcal{A}} (\mu_{i,a}^t - \mu_{i,a}^*) \mathbf{E} [\Delta_{i,a}^{t+1} | \mathcal{F}_t] &= \sum_{a \in \mathcal{A}} \sum_{i \in \mathcal{S}} (\mu_{i,a}^t - \mu_{i,a}^*) [(\gamma P_a \mathbf{v}^t - \mathbf{v}^t + \mathbf{r}_a)_i - M] \\ &= \sum_{a \in \mathcal{A}} (\mu_a^t - \mu_a^*)^\top ((\gamma P_a - I)\mathbf{v}^t + \mathbf{r}_a), \end{aligned}$$

where the second equality comes from the fact $\sum_{i \in \mathcal{S}} \sum_{a \in \mathcal{A}} \mu_{i,a}^t = \sum_{i \in \mathcal{S}} \sum_{a \in \mathcal{A}} \mu_{i,a}^* = 1$ (because $\mu^t \in \mathcal{U}_{\theta, \mathbf{q}}$, $\mu^* \in \mathcal{U}_{\theta, \mathbf{q}}$). According to Lemma 4, we also have $\mathbf{E} \left[\sum_{i \in \mathcal{S}} \sum_{a \in \mathcal{A}} \mu_{i,a}^t \left(\Delta_{i,a}^{t+1} \right)^2 | \mathcal{F}_t \right] \leq \frac{4|\mathcal{S}||\mathcal{A}|\beta^2}{(1 - \gamma)^2}$. We apply the preceding two relations and complete the proof. \blacksquare

Proposition 2. The iterates generated by Algorithm 1 satisfy for all t with probability 1 that

$$\mathbf{E} [\|\mathbf{v}^{t+1} - \mathbf{v}^*\|^2 | \mathcal{F}_t] \leq \|\mathbf{v}^t - \mathbf{v}^*\|^2 + 2\alpha(\mathbf{v}^t - \mathbf{v}^*)^\top \left(\sum_{a \in \mathcal{A}} (I - \gamma P_a)^\top \mu_a^t - (1 - \gamma)\mathbf{q} \right) + 4\alpha^2. \quad (11)$$

Proof. We let $\mathbf{v}^{t+1/2}$ be the vector such that

$$\begin{aligned} v_{i_{t+1}}^{t+1/2} &= v_{i_{t+1}}^t - \alpha \left(\frac{(1 - \gamma)q_{i_{t+1}}}{(1 - \theta)\xi_{i_{t+1}}^t + \theta q_{i_{t+1}}} - 1 \right), \\ v_{j_{t+1}}^{t+1/2} &= v_{j_{t+1}}^t - \alpha \gamma, \\ v_i^{t+1/2} &= v_i^t, \quad \text{if } i \notin \{i_{t+1}, j_{t+1}\}. \end{aligned}$$

Then we can verify that $\mathbf{v}^{t+1} = \Pi_{\mathcal{V}} \mathbf{v}^{t+1/2}$, where Π denotes the Euclidean projection. We note that $\mathbf{P}(i_{t+1} = i \mid \mathcal{F}_t) = (1 - \theta)\xi_i^t + \theta q_i = \sum_{a \in \mathcal{A}} \mu_{i,a}^t$, $\mathbf{P}(j_{t+1} = j \mid \mathcal{F}_t) = \sum_{i \in \mathcal{S}} \sum_{a \in \mathcal{A}} \mu_{i,a}^t p_{ij}(a)$. Then we can verify that

$$\mathbf{E} [\mathbf{v}^{t+1/2} - \mathbf{v}^t \mid \mathcal{F}_t] = -\alpha \left((1 - \gamma)\mathbf{q} - \sum_{a \in \mathcal{A}} (I - \gamma P_a)^\top \mu_a^t \right).$$

Also since $\theta \geq 1 - \gamma$ we have $\frac{(1-\gamma)q_{i_{t+1}}}{(1-\theta)\xi_{i_{t+1}}^t + \theta q_{i_{t+1}}} \in [0, 1]$. Then we have $|v_{i_{t+1}}^{t+1/2} - v_{i_{t+1}}^t| < \alpha$ and $|v_{j_{t+1}}^{t+1/2} - v_{j_{t+1}}^t| < \alpha$. Then we can verify that $\|\mathbf{v}^{t+1/2} - \mathbf{v}^t\|^2 \leq 4\alpha^2$ for all t with probability 1. Finally, by using the nonexpansive property of $\Pi_{\mathcal{V}}$ and $\mathbf{v}^* \in \mathcal{V}$, we further obtain

$$\begin{aligned} \mathbf{E} [\|\mathbf{v}^{t+1} - \mathbf{v}^*\|^2 \mid \mathcal{F}_t] &= \mathbf{E} [\|\Pi_{\mathcal{V}} \mathbf{v}^{t+1/2} - \mathbf{v}^*\|^2 \mid \mathcal{F}_t] \\ &\leq \mathbf{E} [\|\mathbf{v}^{t+1/2} - \mathbf{v}^*\|^2 \mid \mathcal{F}_t] \\ &= \|\mathbf{v}^t - \mathbf{v}^*\|^2 + 2(\mathbf{v}^t - \mathbf{v}^*)^\top \mathbf{E} [\mathbf{v}^{t+1/2} - \mathbf{v}^t \mid \mathcal{F}_t] + \mathbf{E} [\|\mathbf{v}^{t+1/2} - \mathbf{v}^t\|^2 \mid \mathcal{F}_t], \\ &\leq \|\mathbf{v}^t - \mathbf{v}^*\|^2 + 2\alpha(\mathbf{v}^t - \mathbf{v}^*)^\top \left(\sum_{a \in \mathcal{A}} (I - \gamma P_a)^\top \mu_a^t - (1 - \gamma)\mathbf{q} \right) + 4\alpha^2, \end{aligned}$$

for all t with probability 1. ■

Proposition 3. *We define for short that*

$$\mathcal{E}^t = \Phi(\mu^t) + \frac{(1 - \gamma)^2}{|\mathcal{S}|} \|\mathbf{v}^t - \mathbf{v}^*\|^2$$

and

$$\mathcal{G}^t = \sum_{a \in \mathcal{A}} (\mu_a^t)^\top ((I - \gamma P_a)\mathbf{v}^* - \mathbf{r}_a) = \sum_{i \in \mathcal{S}} \sum_{a \in \mathcal{A}} \mu_{i,a}^t (\mathbf{v}^* - \gamma P_a \mathbf{v}^* - \mathbf{r}_a)_i.$$

Let $\alpha = \frac{|\mathcal{S}|}{2(1-\gamma)^2} \beta$. The iterates generated by Algorithm 1 satisfy for all t with probability 1 that

$$\mathbf{E} [\mathcal{E}^{t+1} \mid \mathcal{F}_t] \leq \mathcal{E}^t - \beta \mathcal{G}^t + \beta^2 \frac{2|\mathcal{S}|(|\mathcal{A}| + 1)}{(1 - \gamma)^2}. \quad (12)$$

Proof. Let $\alpha = \frac{|\mathcal{S}|}{2(1-\gamma)^2} \beta$. We multiply (11) with $\frac{(1-\gamma)^2}{|\mathcal{S}|}$ and takes its sum with (10), obtaining

$$\begin{aligned} \mathbf{E} [\mathcal{E}^{t+1} \mid \mathcal{F}_t] &\leq \mathcal{E}^t + \beta^2 \frac{2|\mathcal{S}||\mathcal{A}| + |\mathcal{S}|}{(1 - \gamma)^2} \\ &\quad + \beta \left(\sum_{a \in \mathcal{A}} (\mu_a^t - \mu_a^*)^\top ((\gamma P_a - I)\mathbf{v}^t + \mathbf{r}_a) + (\mathbf{v}^t - \mathbf{v}^*)^\top \left(\sum_{a \in \mathcal{A}} (I - \gamma P_a)^\top \mu_a^t - (1 - \gamma)\mathbf{q} \right) \right). \end{aligned}$$

We have

$$\begin{aligned} &\sum_{a \in \mathcal{A}} (\mu_a^t - \mu_a^*)^\top ((\gamma P_a - I)\mathbf{v}^t + \mathbf{r}_a) + (\mathbf{v}^t - \mathbf{v}^*)^\top \left(\sum_{a \in \mathcal{A}} (I - \gamma P_a)^\top \mu_a^t - (1 - \gamma)\mathbf{q} \right) \\ &= \sum_{a \in \mathcal{A}} (\mu_a^t - \mu_a^*)^\top ((\gamma P_a - I)\mathbf{v}^t + \mathbf{r}_a) + (\mathbf{v}^t - \mathbf{v}^*)^\top \sum_{a \in \mathcal{A}} (I - \gamma P_a)^\top (\mu_a^t - \mu_a^*) \\ &= \sum_{a \in \mathcal{A}} (\mu_a^t - \mu_a^*)^\top ((\gamma P_a - I)\mathbf{v}^* + \mathbf{r}_a) \quad (\text{by the dual feasibility of } \mu^*) \\ &= \sum_{a \in \mathcal{A}} (\mu_a^t)^\top ((\gamma P_a - I)\mathbf{v}^* + \mathbf{r}_a) \quad (\text{by the linear complementarity condition for } \mathbf{v}^*, \mu^*) \end{aligned}$$

where the second equality uses the dual feasibility of μ^* that $\sum_{a \in \mathcal{A}} (I - \gamma P_a)^\top \mu_a^* = (1 - \gamma) \mathbf{q}$ and the fourth equality uses the complementary condition $\mu_{a,i}^* ((\gamma P_a - I) \mathbf{v}^* + \mathbf{r}_a)_i = 0$ for all $i \in \mathcal{S}, a \in \mathcal{A}$. Combining the preceding relations, we obtain (12). ■

B Proofs of Main Theorems

Proof of Theorem 1. We claim that $\mathcal{E}^1 \leq \log(|\mathcal{S}||\mathcal{A}|) + 1$. To see this, we note that λ^1 and π_i^1 's are uniform distributions (according to Step 3 of Algorithm 1). Therefore we have $D_{KL}(\lambda^* || \lambda^1) \leq \log(SA)$ and $D_{KL}(\pi_i^* || \pi_i^1) \leq \log(S)$ for i , and $\|\mathbf{v}^t - \mathbf{v}^*\|^2 \leq \frac{S}{(1-\gamma)^2}$ for all t . Then we have $\mathcal{E}^1 \leq (1 - \theta) D_{KL}(\lambda^* || \lambda^1) + \theta \sum_{i \in \mathcal{S}} q_i D_{KL}(\pi_i^* || \pi_i^1) + \frac{(1-\gamma)^2}{|\mathcal{S}|} \|\mathbf{v}^1 - \mathbf{v}^*\|^2 \leq \log(|\mathcal{S}||\mathcal{A}|) + 1$.

We rearrange the terms of (12) and obtain

$$\mathcal{G}^t \leq \frac{1}{\beta} (\mathcal{E}^t - \mathbf{E} [\mathcal{E}^{t+1} | \mathcal{F}_t]) + \frac{2\beta|\mathcal{S}|(|\mathcal{A}| + 1)}{(1 - \gamma)^2}.$$

Summing over $t = 1, \dots, T$ and taking average, we have

$$\begin{aligned} \mathbf{E} \left[\sum_{t=1}^T \mathcal{G}^t \right] &\leq \frac{1}{\beta} \sum_{t=1}^T (\mathbf{E} [\mathcal{E}^t] - \mathbf{E} [\mathcal{E}^{t+1}]) + \frac{2\beta|\mathcal{S}|(|\mathcal{A}| + 1)T}{(1 - \gamma)^2} \\ &= \frac{\mathbf{E} [\mathcal{E}^1] - \mathbf{E} [\mathcal{E}^T]}{\beta} + \frac{2\beta|\mathcal{S}|(|\mathcal{A}| + 1)T}{(1 - \gamma)^2} \\ &\leq \frac{1}{\beta} (\log(|\mathcal{S}||\mathcal{A}|) + 1) + \frac{2\beta|\mathcal{S}|(|\mathcal{A}| + 1)T}{(1 - \gamma)^2}. \end{aligned}$$

where the inequality is based on the fact $\mathcal{E}^1 \leq \log(|\mathcal{S}||\mathcal{A}|) + 1$ and $\mathcal{E}^T \geq 0$. Therefore by taking $\beta = (1 - \gamma) \sqrt{\frac{\log(|\mathcal{S}||\mathcal{A}| + 1)}{2|\mathcal{S}||\mathcal{A}|T}}$, we obtain $\mathbf{E} \left[\frac{1}{T} \sum_{t=1}^T \mathcal{G}^t \right] \leq \frac{\sqrt{2|\mathcal{S}|(|\mathcal{A}| + 1)(\log(|\mathcal{S}||\mathcal{A}| + 1))}}{(1 - \gamma)\sqrt{T}}$. ■

Proof of Theorem 2 We first show that $\mu^* \in \mathcal{U}_{\theta, \mathbf{q}}$ when $\theta = 1 - \gamma$, $\mathbf{q} = \frac{1}{S} \mathbf{e}$. To see this, we note the dual feasibility constraint $\sum_{a \in \mathcal{A}} (I - \gamma P_a)^\top \mu_a^* = (1 - \gamma) \mathbf{q}$, which implies a lower bound to the dual variable μ , given by $\sum_{a \in \mathcal{A}} \mu_a^* \geq (1 - \gamma) \mathbf{q}$. Therefore $\mu^* \in \mathcal{U}_{\theta, \mathbf{q}}$ and the assumption of Theorem 1 is satisfied.

In the following, we show that the duality gap for the bilinear saddle point problem gives a bound on the efficiency loss of the randomized policy $\hat{\pi}$. Denote for short $\hat{\mu} = \frac{1}{T} \sum_{t=1}^T \mu^t$. We have

$$\begin{aligned} \frac{1}{T} \sum_{t=1}^T \mathcal{G}^t &= \sum_{a \in \mathcal{A}, i \in \mathcal{S}} \hat{\mu}_{i,a} (\mathbf{v}^* - \gamma P_a \mathbf{v}^* - \mathbf{r}_a)_i \\ &\geq (1 - \gamma) \sum_{i \in \mathcal{S}} q_i \sum_{a \in \mathcal{A}} \hat{\pi}_{a,i} (\mathbf{v}^* - \gamma P_a \mathbf{v}^* - \mathbf{r}_a)_i \\ &= (1 - \gamma) \sum_{i \in \mathcal{S}} q_i (\mathbf{v}^* - \gamma P^{\hat{\pi}} \mathbf{v}^* - \mathbf{r}^{\hat{\pi}})_i, \end{aligned}$$

where $\mathbf{r}^{\hat{\pi}}$ denotes the vector with $r_i^{\hat{\pi}} = \sum_{a \in \mathcal{A}} \hat{\pi}_i(a) \sum_{j \in \mathcal{S}} p_{ij}(a) r_{ij}(a)$. We note the Bellman equation for a fixed policy $\hat{\pi}$ given by $\mathbf{v}^{\hat{\pi}} = \gamma P^{\hat{\pi}} \mathbf{v}^{\hat{\pi}} + \mathbf{r}^{\hat{\pi}}$. Because \mathbf{v}^* is the optimal value vector for the DMDP, we have $(\mathbf{v}^* - \gamma P_a \mathbf{v}^* - \mathbf{r}_a)_i \geq 0$ for all $i \in \mathcal{S}$. It follows that $(\mathbf{v}^* - \gamma P^{\hat{\pi}} \mathbf{v}^* - \mathbf{r}^{\hat{\pi}})_i \geq 0$ for $i \in \mathcal{S}$, therefore

$$0 \leq \mathbf{v}^* - \gamma P^{\hat{\pi}} \mathbf{v}^* - \mathbf{r}^{\hat{\pi}} = \mathbf{v}^* - \gamma P^{\hat{\pi}} \mathbf{v}^* - (\mathbf{v}^{\hat{\pi}} - \gamma P^{\hat{\pi}} \mathbf{v}^{\hat{\pi}}) = (I - \gamma P^{\hat{\pi}})(\mathbf{v}^* - \mathbf{v}^{\hat{\pi}}).$$

Using the fact $\mathbf{q} = \frac{1}{|\mathcal{S}|}\mathbf{e}$, we further have $\frac{1}{T} \sum_{t=1}^T \mathcal{G}^t = (1 - \gamma)(\mathbf{q})^\top (I - \gamma P^{\hat{\pi}})(\mathbf{v}^* - \mathbf{v}^{\hat{\pi}}) \geq \frac{1 - \gamma}{|\mathcal{S}|} \|(I - \gamma P^{\hat{\pi}})(\mathbf{v}^* - \mathbf{v}^{\hat{\pi}})\|_\infty$. We use the triangle inequality and the matrix norm inequality $\|Ax\|_\infty \leq \|A\|_\infty \|x\|_\infty$ to obtain $\|(I - \gamma P^{\hat{\pi}})(\mathbf{v}^* - \mathbf{v}^{\hat{\pi}})\|_\infty \geq \|\mathbf{v}^* - \mathbf{v}^{\hat{\pi}}\|_\infty - \|\gamma P^{\hat{\pi}}(\mathbf{v}^* - \mathbf{v}^{\hat{\pi}})\|_\infty \geq \|\mathbf{v}^* - \mathbf{v}^{\hat{\pi}}\|_\infty - \|\gamma P^{\hat{\pi}}\|_\infty \|\mathbf{v}^* - \mathbf{v}^{\hat{\pi}}\|_\infty = (1 - \gamma) \|\mathbf{v}^* - \mathbf{v}^{\hat{\pi}}\|_\infty$. It follows that $\left(\frac{1}{T} \sum_{t=1}^T \mathcal{G}^t\right) \geq \frac{(1 - \gamma)^2}{S} \|\mathbf{v}^* - \mathbf{v}^{\hat{\pi}}\|_\infty$. Now we apply Theorem 1 and the Markov inequality to obtain $\|\mathbf{v}^* - \mathbf{v}^{\hat{\pi}}\|_\infty \leq \frac{S}{(1 - \gamma)^2} \left(\frac{1}{T} \sum_{t=1}^T \mathcal{G}^t\right) \leq \mathcal{O}\left(\frac{S}{(1 - \gamma)^2}\right) \mathbf{E} \left[\left(\frac{1}{T} \sum_{t=1}^T \mathcal{G}^t\right) \right]$ with probability at least $2/3$. Finally, we note that $v_i^* \geq \frac{1}{2(1 - \gamma)}$ for all i . Therefore $\mathbf{v}^\pi \geq \mathbf{v}^* - \frac{\epsilon}{2(1 - \gamma)}\mathbf{e} \geq (1 - \epsilon)\mathbf{v}^*$ with probability $2/3$ when $\mathbf{E} \left[\left(\frac{1}{T} \sum_{t=1}^T \mathcal{G}^t\right) \right] \leq \mathcal{O}\left(\frac{\epsilon(1 - \gamma)}{|\mathcal{S}|}\right)$, which holds if we let $T = \Omega\left(\frac{|\mathcal{S}|^3 |\mathcal{A}| \log(|\mathcal{S}| |\mathcal{A}|)}{(1 - \gamma)^4 \epsilon^2}\right)$. \blacksquare

Proof of Theorem 3 We first verify that $\mu^* \in \mathcal{U}_{\theta, \mathbf{q}}$ with the given vector \mathbf{q} and $\theta = 1 - \gamma + \gamma \frac{c_1}{c_2}$. We note that all eigenvalues of a probability transition matrix P^π belong to the unit circle and $\gamma \in (0, 1)$, therefore the eigenvalues $I - \gamma P^\pi$ belong to the positive half plane. As a result, the matrix $I - \gamma P^\pi$ is invertible for all π , including $I - \gamma P^*$. We have $\sum_{a \in \mathcal{A}} \mu_a^* = (I - \gamma(P^*)^\top)^{-1} (1 - \gamma)\mathbf{q}$, therefore

$$\begin{aligned} \sum_{a \in \mathcal{A}} \mu_a^* &= (1 - \gamma) \left(\sum_{k=0}^{\infty} (\gamma P^*)^k \right)^\top \mathbf{q} = (1 - \gamma)\mathbf{q} + (1 - \gamma) \left(\sum_{k=1}^{\infty} (\gamma P^*)^k \right)^\top \mathbf{q} \\ &\geq (1 - \gamma)\mathbf{q} + \frac{1}{c_2} (1 - \gamma) \sum_{k=1}^{\infty} \left((\gamma P^*)^k \right)^\top \nu^* \\ &= (1 - \gamma)\mathbf{q} + \frac{1}{c_2} (1 - \gamma) \left(\sum_{k=1}^{\infty} \gamma^k \right) \nu^* \\ &\geq (1 - \gamma)\mathbf{q} + \frac{c_1}{c_2} \gamma \mathbf{q} \\ &\geq \left(1 - \gamma + \gamma \frac{c_1}{c_2} \right) \mathbf{q}. \end{aligned}$$

As a result, we have verified $\mu^* \in \mathcal{U}_{\theta, \mathbf{q}}$ and the assumption and results of Theorem 1 hold.

When Algorithm 1 is applied with input $\theta = 1 - \gamma + \gamma \frac{c_1}{c_2}$. By the nature of the algorithm, we have $\sum_{a \in \mathcal{A}} \hat{\mu}_a \geq \theta \mathbf{q}$. Then we have and $c_1 \mathbf{q} \geq \nu^\pi \geq c_2 \mathbf{q}$ for all π , we have

$$\begin{aligned} \left(\frac{1}{T} \sum_{t=1}^T \mathcal{G}^t \right) &= \sum_{a \in \mathcal{A}, i \in \mathcal{S}} \hat{\mu}_{a,i} (\mathbf{v}^* - \gamma P_a \mathbf{v}^* - \mathbf{r}_a)_i \geq \left(1 - \gamma + \gamma \frac{c_1}{c_2} \right) \sum_{i \in \mathcal{S}} q_i \sum_{a \in \mathcal{A}} \hat{\pi}_{i,a} (\mathbf{v}^* - \gamma P_a \mathbf{v}^* - \mathbf{r}_a)_i \\ &= \left(1 - \gamma + \gamma \frac{c_1}{c_2} \right) \sum_{i \in \mathcal{S}} q_i (\mathbf{v}^* - \gamma P^{\hat{\pi}} \mathbf{v}^* - \mathbf{r}^{\hat{\pi}})_i \\ &\geq \left(1 - \gamma + \gamma \frac{c_1}{c_2} \right) \sum_{i \in \mathcal{S}} \frac{1}{c_2} \nu_i^{\hat{\pi}} (\mathbf{v}^* - \gamma P^{\hat{\pi}} \mathbf{v}^* - \mathbf{r}^{\hat{\pi}})_i \\ &= \left(1 - \gamma + \gamma \frac{c_1}{c_2} \right) \frac{1}{c_2} \left(\nu^{\hat{\pi}} \right)^\top (I - \gamma P^{\hat{\pi}}) (\mathbf{v}^* - \mathbf{v}^{\hat{\pi}}) \\ &= \left(1 - \gamma + \gamma \frac{c_1}{c_2} \right) \frac{1}{c_2} (1 - \gamma) \left(\nu^{\hat{\pi}} \right)^\top (\mathbf{v}^* - \mathbf{v}^{\hat{\pi}}) \\ &\geq \frac{c_1^2}{c_2^2} (1 - \gamma) \mathbf{q}^\top (\mathbf{v}^* - \mathbf{v}^{\hat{\pi}}). \end{aligned}$$

It follows that $\mathbf{q}^\top(\mathbf{v}^* - \mathbf{v}^\pi) \leq \frac{c_2^2}{c_1^2(1-\gamma)} \frac{1}{T} \sum_{t=1}^T \mathcal{G}^t$. By using a similar analysis as in the proof of Theorem 2, we finish the proof. \blacksquare

C Proofs of Corollaries

We aim to run Algorithm 1 for multiple trials to obtain an ϵ -optimal policy with probability arbitrarily close to 1. This requires us be able to evaluate multiple candidate policies and select the best one out of many. We show that it is possible to approximately evaluate any policy π within ϵ -precision in running time $\tilde{\mathcal{O}}(\frac{1}{\epsilon^2})$ (see Lemma 5 in Appendix C).

Lemma 5 (Approximate Policy Evaluation). *Suppose we are given a DMDP tuple $\mathcal{M} = (\mathcal{S}, \mathcal{A}, \mathcal{P}, \mathbf{r}, \gamma)$, a fixed policy π , and an initial distribution \mathbf{q} . Suppose that a state transition of the DMDP can be sampled in $\tilde{\mathcal{O}}(1)$ time, there exists an algorithm that outputs an approximate value \bar{Y} such that $\mathbf{q}^\top \mathbf{v}^\pi - \epsilon \leq \bar{Y} \leq \mathbf{q}^\top \mathbf{v}^\pi$ with probability at least $1 - \delta$ in running time $\tilde{\mathcal{O}}(\frac{1}{\epsilon^2(1-\gamma)^2} \log(\frac{1}{\delta}))$.*

Proof. For a given policy π , we simulate the Markov decision process under policy π from the initial distribution \mathbf{q} for n transitions and calculate the cumulative discounted reward Y . We repeat the simulation for K independent times and return the average cumulative reward $\bar{Y} = \frac{1}{K}(Y_1 + \dots + Y_K)$.

We observe that the unknown value $\mathbf{q}^\top \mathbf{v}^\pi$ is the expectation of the infinite cumulative discounted reward. We pick n sufficiently large such that expected n -period cumulative discounted reward is sufficiently close to $\mathbf{q}^\top \mathbf{v}^\pi$. Since $r_{ij}(a) \in [\frac{1}{2}, 1]$ for i, j, a , the cumulative discounted reward starting from the $(n+1)$ th period is bounded by $\sum_{t=n}^{\infty} \gamma^t = \frac{\gamma^n}{1-\gamma}$ with probability 1. In particular, we pick n such that $\frac{\epsilon}{2} = \frac{\gamma^n}{1-\gamma}$, which suggests that $n = \left(\log_\gamma(\frac{\epsilon(1-\gamma)}{2})\right) = \tilde{\mathcal{O}}(1)$. Therefore we obtain

$$\mathbf{q}^\top \mathbf{v}^\pi - \frac{\epsilon}{2} \leq \mathbf{E}[Y_1] \leq \mathbf{q}^\top \mathbf{v}^\pi.$$

Note that Y_1, \dots, Y_K are i.i.d. random variables and $Y_k \in [0, \frac{1}{1-\gamma}]$ for all k . By using the Azuma-Hoeffding inequality, we obtain that $\bar{Y} = \frac{1}{K} \sum_{t=1}^K Y_t$ satisfies for any $\epsilon > 0$ that

$$\mathbf{P}(|\bar{Y} - \mathbf{E}[Y_1]| \geq \epsilon) \leq 2 \exp\left(-\frac{\epsilon^2 K (1-\gamma)^2}{2}\right).$$

By letting $\epsilon = \frac{\epsilon}{2}$ and $K = \mathcal{O}(\frac{1}{\epsilon^2(1-\gamma)^2} \log(1/\delta))$, we obtain that $|\bar{Y} - \mathbf{E}[Y_1]| \leq \frac{\epsilon}{2}$ with probability at least $1 - \delta$. It follows that

$$\mathbf{P}\left(\mathbf{q}^\top \mathbf{v}^\pi - \epsilon \leq \bar{Y} \leq \mathbf{q}^\top \mathbf{v}^\pi\right) \geq 1 - \delta.$$

The total number of sample state transitions is $K \cdot n = \tilde{\mathcal{O}}(\frac{1}{\epsilon^2(1-\gamma)^2} \log(\frac{1}{\delta}))$. \blacksquare

Now we prove that by repeatedly running Algorithm 1 and using approximate policy evaluation, one can compute a near-optimal policy with probability arbitrarily close to 1.

Proof of Corollaries 1 and 2. Note that $\mathbf{q}^\top \mathbf{v}^* \geq \frac{1}{2(1-\gamma)}$ (because $r_{ij}(a) \in [1/2, 1]$ for all i, j, a).

We describe an approach that runs Algorithm 1 for multiple times in order to achieve an ϵ -optimal policy with probability $1 - \delta$:

1. We first run Algorithm 1 for K independent trials with precision parameter $\frac{\epsilon}{2}$, and we denote the output policies by $\pi^{(1)}, \dots, \pi^{(K)}$. The total running time is $\mathcal{O}(|\mathcal{S}|^2 |\mathcal{A}|) + K \cdot T_{\frac{\epsilon}{2}}$, where $T_{\frac{\epsilon}{2}}$ is the time complexity for each run of Algorithm 1. According to either Theorem 1 or 2, each trial generates an $\epsilon/2$ -optimal policy with probability at least $2/3$.

2. For each output policy $\pi^{(k)}$, we conduct approximate value evaluation and obtain an approximate evaluation $\bar{Y}^{(k)}$ with precision level $\frac{\epsilon}{4(1-\gamma)}$ and fail probability $\frac{\delta}{2K}$. According to Lemma 5, we have

$$\bar{Y}^{(k)} - \mathbf{q}^\top \mathbf{v}^{\pi^{(k)}} \in \left[-\frac{\epsilon}{4(1-\gamma)}, 0\right] \subset \left[-\frac{\epsilon}{2} \mathbf{q}^\top \mathbf{v}^*, 0\right],$$

with probability at least $1 - \frac{\delta}{2K}$, and this step takes $K \cdot \tilde{\mathcal{O}}(\frac{1}{\epsilon^2} \log(\frac{K}{\delta}))$ running time.

3. Output $\hat{\pi} = \pi^{(k^*)}$ such that $k^* = \operatorname{argmax}_{k=1, \dots, K} \bar{Y}^{(k)}$. This step takes $\mathcal{O}(K)$ running time.

Now we verify that $\hat{\pi}$ is indeed near-optimal with probability at least $1 - \delta$ when K is chosen appropriately. Let $\mathcal{K} = \left\{k \mid \mathbf{q}^\top \mathbf{v}^{\pi^{(k)}} \geq (1 - \frac{\epsilon}{2}) \mathbf{q}^\top \mathbf{v}^*\right\}$, which can be interpreted as indices of successful trails of Algorithm 1. Consider the event where $\mathcal{K} \neq \emptyset$ and all policy evaluation errors belong to the small interval $[-\frac{\epsilon}{2} \mathbf{q}^\top \mathbf{v}^*, 0]$. In this case, we have $\bar{Y}^{(k)} \leq \mathbf{q}^\top \mathbf{v}^{\pi^{(k)}}$ for all k and $\bar{Y}^{(k)} \geq (1 - \epsilon) \mathbf{q}^\top \mathbf{v}^*$ if $k \in \mathcal{K}$. As a result, the output policy which has the largest value of $\bar{Y}^{(k)}$ must be ϵ -optimal. We use the union bound to obtain

$$\begin{aligned} \mathbf{P}\left(\mathbf{q}^\top \mathbf{v}^{\hat{\pi}} < (1 - \epsilon) \mathbf{q}^\top \mathbf{v}^*\right) &\leq \mathbf{P}\left(\{\mathcal{K} = \emptyset\} \cup \left\{\exists k : \bar{Y}^{(k)} - \mathbf{q}^\top \mathbf{v}^{\pi^{(k)}} \notin \left[-\frac{\epsilon}{2} \mathbf{q}^\top \mathbf{v}^*, 0\right]\right\}\right) \\ &\leq \mathbf{P}(\mathcal{K} = \emptyset) + \mathbf{P}\left(\exists k : \bar{Y}^{(k)} - \mathbf{q}^\top \mathbf{v}^{\pi^{(k)}} \notin \left[-\frac{\epsilon}{2} \mathbf{q}^\top \mathbf{v}^*, 0\right]\right) \\ &\leq \prod_{k=1}^K \mathbf{P}\left(\mathbf{q}^\top \mathbf{v}^{\pi^{(k)}} < (1 - \frac{\epsilon}{2}) \mathbf{q}^\top \mathbf{v}^*\right) + \sum_{k=1}^K \mathbf{P}\left(\bar{Y}^{(k)} - \mathbf{q}^\top \mathbf{v}^{\pi^{(k)}} \notin \left[-\frac{\epsilon}{2} \mathbf{q}^\top \mathbf{v}^*, 0\right]\right) \\ &\leq (1/3)^K + K \cdot \frac{\delta}{2K}. \end{aligned}$$

By choosing $K = \Omega(\log(1/\delta))$, we obtain $\mathbf{P}(\mathbf{q}^\top \mathbf{v}^{\hat{\pi}} < (1 - \epsilon) \mathbf{q}^\top \mathbf{v}^*) \leq \delta$. Then the output policy is ϵ -optimal with probability at least $1 - \delta$.

According to Section 4, preprocessing of Algorithm 1 takes $\tilde{\mathcal{O}}(|\mathcal{S}|^2 |\mathcal{A}|)$ arithmetic operations and each iteration takes $\tilde{\mathcal{O}}(1)$ arithmetic operations. The total running time is $\tilde{\mathcal{O}}(|\mathcal{S}|^2 |\mathcal{A}| + T_{\frac{\epsilon}{2}} \log \frac{1}{\delta} + \frac{1}{\epsilon^2} (\log \frac{1}{\delta})^2)$. \blacksquare